Infotech@Aerospace 2011
29 - 31 March 2011, St. Louis, Missouri

AIAA 2011-1406

Metrics and Analysis Methods for Evaluation of a Real-Time Automated Separation Assurance System

Scott Sahlman*
University of California Santa Cruz, Moffett Field, CA, 94035

A system of metrics and analysis plots has been developed in order to evaluate the effectiveness and accuracy of trajectory prediction and automated conflict prediction and resolution algorithms. This research is conducted using a real-time air traffic management system to perform experimental runs using both live and simulated air traffic scenarios. The resulting data from these runs needs to be analyzed both to evaluate the performance of the software and to create objective metrics and plots for use in research studies. Specific analysis methods have been developed to assess data runs based on losses of legal separation between aircraft, conflict detections, trajectory prediction error, flight amendments, level-offs, and fuel burn.

I. Introduction

NASA is developing tools and concepts to assist air traffic controllers in ensuring safe and efficient operations under increasing traffic demand. This work is divided into specialized tasks to focus on addressing specific air traffic management (ATM) needs. One such task is Separation Assurance (SA) research whose primary goals are to study automated algorithms for ensuring adequate separation between aircraft and to improve traffic flow through Air Route Traffic Control Center (ARTCC), or Center, airspace by using efficient, conflict-free flight paths.[1] In this research, the Center TRACON Automation System (CTAS)[2-4] is an important ATM tool with capabilities for automated conflict detection and resolution used as an experimental environment for comparing actual and simulated air traffic scenarios. The evaluation of the performance of this system under real-time traffic conditions is critical for proving the effectiveness of automated separation assurance algorithms[5] as well as trajectory prediction algorithms for possible future use in a live air traffic control environment.

A fundamental component of this research[1,5,6] is an ongoing effort to design and develop a set of objective metrics and associated analysis plots used to evaluate the performance of an automated SA system, compare the performance of an automated SA system against the performance of an actual system using common metrics, and show how various parameters, events, or trends relate to the performance of an experimental data run. This is seldom a straightforward matter because data, especially live data, are often noisy, vague, or irregular and may require approximations, assumptions, or statistical analysis to produce useful results. At the same time, the particular methods used to define specific metrics may need to be taken into consideration when using or presenting the results of such analysis. These analysis results serve to support two main objectives: continued development and testing of the software and production of objective metrics or plots for inclusion in research publications and presentations. For example, a standard analysis suite can be used for regular regression testing of the research software, or an analysis routine could be designed to verify that a new feature is working as specified. Research papers or presentations often require analyses that show that the software is behaving within certain parameters or how the performance varies when using different data samples, input variables, or software features. These two objectives are not mutually exclusive, and often the same analysis methods can be used for both. Ultimately the intent is to support both researchers and developers with the creation of tools to help analyze data inputs, software algorithms, and run outcomes.

This paper describes the approach to the design and implementation of SA analysis metrics and also demonstrates some of the metrics currently used in the evaluation of trajectory prediction and automated conflict detection and resolution algorithms. The next section begins by giving some background on the basic experimental setup used in this SA research accompanied by an explanation of the general approach used when designing new analyses. The tools used for creating analyses are then described followed by a brief discussion of the methods used

* Senior Programmer/Analyst, NASA Ames Research Center, Mail Stop 210-217, AIAA Member.

for verification of new analyses. Several examples of metrics and plots created for SA analysis are then detailed. The paper finishes with some concluding remarks.

II. SA Experiment Background

The types of analysis performed are going to depend on the experiment on which they are based. For this research, CTAS is used to study various air traffic scenarios usually in Fort Worth Center, known as ZFW Center. CTAS is a Center-based, real-time ATM system with 4-D (space and time) trajectory prediction and decision support features such as aircraft trial planning and automated conflict prediction and resolution algorithms.[2-5] CTAS can be used to display, record, and replay data from live air traffic radar feeds, and it supports the ability to simulate control of aircraft in Center airspace. Because live air traffic data already contain resolutions and amendments made by real-world controllers, CTAS is used to create simulated air traffic scenarios where the software can assume control of aircraft and the automation can be allowed to resolve conflicts according to SA algorithms.

All data runs are based on recordings of live air traffic radar data. These recordings are often about three to four hours in length, but longer runs such as eight hours may also be used. The two main categories of runs are live data runs and simulation data runs. Live runs, also called shadow runs, use original recordings of live air traffic radar feeds to create additional archived run data such as 4-D trajectory predictions and conflict detections.

For simulation runs CTAS is run in a special mode, called "feedback mode," that allows it to assume control of an aircraft when certain criteria are met. In feedback mode, a live recording is played back to provide initial traffic data. Aircraft that meet specific conditions set within CTAS, the primary ones being a minimum altitude and a minimum amount of track data, are then "taken over" by CTAS and flown based on 4-D trajectory data fed back by the CTAS flight prediction algorithms. Although aircraft trajectories continue to be recomputed by CTAS every twelve seconds, only the first trajectory computed when the aircraft is taken over or the first trajectory computed in response to a flight amendment is used to create the aircraft's path. This creates a more consistent and realistic flight path for the aircraft. Once an aircraft has been taken over in feedback mode, it is simulation controlled and the original, live aircraft data is no longer used.

Simulation runs are further divided into two types: open-loop runs and closed-loop runs. In an open-loop run, the automated conflict resolution algorithms are deactivated and aircraft are allowed to fly according to their initial flight plans, specifically those flight plans that are active when the aircraft is taken over. No traffic management controls are applied to aircraft once they are simulation controlled, and they are allowed to violate aircraft separation criteria with other aircraft. Open-loop runs are useful as a baseline comparison case showing what needs to be solved in a given scenario, and the number of resulting losses of separation that are detected provides a metric for the level of complexity of the traffic. In this analysis a loss of separation is generally considered as a failure to maintain a separation greater than 5 nautical miles horizontally and 1000 feet vertically.

In closed-loop runs the automatic SA detection and resolution algorithms are activated and implemented by the CTAS simulation control. Closed-loop runs can be performed using different operational modes such as utilizing direct flights to downstream fixes (referred to as direct-to routes),[4] varying aircraft weight or response time to amendments, or setting different aircraft separation specifications. Some of these modes are intended to make the simulated output look more like real traffic data, and other modes are used to demonstrate the effects of different types of automation in the traffic.

For all the types of runs that are performed, another important distinction in SA research is the time domain for conflict predictions. There are two main ranges, called strategic and tactical, determined by the predicted time to first loss for a conflict, though the defined boundaries of these ranges can vary. In this research strategic SA is generally considered to apply to conflict detections 3 to 8 minutes before first loss of separation, and tactical SA is considered to apply to cases when detections are 3 minutes or less before first loss (although in Ref. 7, 2 minutes is used as this cutoff time). A distinction is needed because different rules and solutions apply depending on the time-to-go before a predicted loss of separation. This research is currently focused mainly on strategic range algorithms and solutions, although there is an algorithm for computing tactical range solutions known as Tactical Separation-Assured Flight Environment (TSAFE)[8] that is sometimes run in conjunction with strategic range solutions.

III. Analysis Design Approach

When a new type of analysis, such as a new plot, is desired, one of the first considerations is whether or not all necessary data are available within existing CTAS or external data files. Although CTAS archives an extensive amount of data from a run, sometimes, new values are needed or existing data need to be enhanced. In these cases a

software modification must be made to CTAS to produce the necessary data. Other times a particular analysis computation might require some additional data outside of CTAS to be obtained such as the Base of Aircraft Data (BADA).[9]

Once the required data have been identified, another important consideration is whether the new analysis requirement is a minor enhancement to an existing analysis routine, a modification or addition to an existing analysis routine, or a new analysis routine entirely. A minor change to an existing analysis routine might be something as simple as changing the units on an axis, or adding a new bar to a graph or columns to a table and is generally the fastest and easiest type of change. If an existing analysis routine uses the same source data or performs similar enough processing as a new analysis requirement, then sometimes the new requirement can be implemented by reusing an existing analysis routine and adding additional processing, additional source data, additional output, or some combination thereof. Some examples of this type of change might be adding a new plot using the same source data, modifying an existing algorithm to produce new results and displaying them on an existing plot, or modifying an existing algorithm to process new source data to produce a new table of output data. This type of change typically ranges from easy to moderately difficult depending on how extensive the changes are to an existing analysis routine. Finally if a new analysis routine is sufficiently different from existing analysis routines, then a new routine will have to be created. Whenever possible, pieces of existing analysis routines such as functions, algorithms, or plot formatting will be reused in the new routine. New routines can range anywhere from relatively easy to difficult depending on the complexity, and could take anywhere from a day to a month to implement.

When either creating or modifying an analysis routine, a couple of guiding principles are often used. The first is simplicity. The basic ideas and algorithms behind an analysis routine should be easy to describe, even if the actual algorithms or computations to create the routine are complex. The other is clarity, and this can be more challenging to achieve. Output from an analysis, such as a plot, should be fairly easy to look at and understand. Related results might be grouped together or colored similarly while other results may be more separate or use different colorations, patterns or line styles. Care should be taken when adding more data or results to the same plot, as too much data can be visually overwhelming. Sometimes the additional data elements can be accommodated by variations in color, style, size, or location. Other times additional data can be simplified to show just key points such as average values or starting points or times of specific occurrences. Sometimes the best approach is just to create multiple outputs. Finally, the colors red and green when used in a plot can imply their own meaning and should used carefully. Green often has the connotation of being good, safe, or nominal, while red often connotes dangerous, bad or undesirable results. Ref. 10 discusses some of these types of issues and gives examples of good and bad uses of colors for visualization purposes.

IV. Analysis Tools

The analysis tool of choice for examples included in this paper is Matlab.[11] Matlab offers a variety of plotting utilities in addition to a multitude of built-in functions to facilitate the rapid prototyping of many types of analysis while also having a robust programming language to support more complex analysis. Some types of processing are better done outside of Matlab, or with existing code, and it can support these types of external calls as well. For example, one shortcoming of Matlab is that it is not efficient at reading in long data files of mixed format. To solve this issue, additional utilities can be written in C, Perl, Java, etc. to process original data sources either before running a Matlab analysis or as part of the Matlab analysis.

The CTAS simulation recording is one such file that contains a large amount of data in various formats. Information is often extracted from it via a parsing utility written in C that produces a structure containing specified data fields for each aircraft. At other times, a Perl script may be used to extract more targeted data from the simulation file. Other text files may be processed in Matlab directly or through external scripts. Some CTAS files are stored in binary format and are accessed using extraction utilities written in C. In other cases there are additional utilities for accessing data from other sources such as airspace data used by CTAS.

V. Verification of Analysis Results

The primary method of verifying analysis results is often by inspection. Results that are missing, in unexpected or invalid ranges, or that seem to show incorrect proportions to other data are often indications of an error in the analysis. Running the same analysis on multiple data sets often reveals unanticipated cases or errors in the analysis algorithms, and can produce output that can be compared among different runs to check for consistency of results. Another approach is to print out additional data or plots to show that analysis results within a run are correct or

consistent, to identify the source of results, or to verify that expected sections of the analysis algorithm are being utilized. Sometimes recomputing a few sample values manually can verify that a computation algorithm of the analysis is working correctly. Finally, Matlab itself has excellent built-in debugging tools that allow an analysis program to be stopped in specific places or under specific conditions to monitor that internal and output values are being computed in an anticipated manner or check whether algorithm execution is passing through anticipated sections of code.

VI. Analysis Metrics

Analysis for SA research is required that, first and foremost, can identify cases where adequate separation is lost between an aircraft pair, known as loss cases, and provide data surrounding this loss to identify various factors that may have contributed to it. Loss cases are usually studied only for simulation runs, as losses in live data are extremely rare. In addition, metrics are also desirable for studying conflict detections, trajectory prediction accuracy, counts of aircraft amendments, frequency of level-offs, and aircraft fuel use. These metrics are used to evaluate individual runs, compare different simulation runs, or compare simulation runs with and live runs.

A. Loss of Separation

In loss of separation analysis the track data for all aircraft in the simulation are examined and aircraft pairs are identified in which a loss of legal separation has occurred. The objective is to study these loss cases to determine the cause of each loss and to identify corrections or improvements to the automation system or software. Although the track data for aircraft is recorded in approximately 12-second intervals, the actual timing for these recordings can vary slightly, and different aircraft may be recorded on different intervals. Thus it is not possible to create precise comparisons of the aircraft using their recorded position and time data. Instead all aircraft positions are linearly interpolated to a uniform and synchronized 12-second interval. This allows for a direct comparison of aircraft positions at a synchronized time.

Loss of separation analysis is applied to both open-loop and closed-loop simulation runs. Open-loop runs are expected to have a large number of losses detected because no conflict resolutions are being applied to the aircraft. Both the count of losses and the loss pairs themselves can be used as a baseline for comparison with the closed-loop case. This comparison between open- and closed-loop cases is only approximate, though, because aircraft paths will vary between the runs and thus the potential conflicts and losses will not be exactly the same. However, within the confines of the airspace around a single Center, many of the same aircraft pairs can be expected to interact in both runs, and the comparison is still useful.

When studying losses from CTAS simulation runs, a raw count of losses is not a sufficient metric. Running CTAS in feedback mode to create a simulation combines live and simulated air traffic because all aircraft must start off using live data before they can be taken over, and a few aircraft may never satisfy the takeover criteria. This creates an environment where the simulation is not in control of all the aircraft and live aircraft are neither aware of the locations of the simulated aircraft nor operating under the same control as the simulated aircraft. Thus it is not uncommon for losses to occur especially in cases where aircraft are first entering the system, and even with ideal conflict resolution algorithms, some losses could still occur in the simulated environment. Because of this, CTAS simulation losses are further categorized to identify critical losses, non-critical losses, and out-of-range losses. These are referred to as Category 1, 2, and 3 losses respectively.

Category 1 losses are characterized generally by losses that occur inside ZFW Center, where the simulation controls at least one of the aircraft for a minimum amount of time, there is a minimum amount of track data for both aircraft, and both aircraft are above an altitude threshold. It also includes any cases that don't clearly fall into the subsequent categories. The expectation is that most Category 1 losses should be resolved by the automation. Cases in Category 2 are typically losses that occur outside ZFW Center that would have been handled by a controller before entering ZFW space, or "pop-up" cases where one or both aircraft have less than three minutes of track data or are controlled by the simulation for less than three minutes. Category 3 losses are defined as those below a certain desired altitude threshold, and are generally of less interest. These Category 3 cases are often due to the problem of arrival aircraft merging together on approach to the meter fix, and are not handled adequately by the current automation[7]. In comparing runs, the Category 1 losses are most important with attention also being placed on Category 2 losses.

Loss cases are analyzed primarily via two methods. First is a simple list of aircraft pairs, called loss pairs, with some additional data about the state of each aircraft. Each loss pair is computed at the point of first loss of separation and listed with the type of each aircraft, the position of each aircraft, the horizontal separation at first loss, the flight category of each (arrival, departure, overflight, etc.), and the time of the loss. In addition there may be details on

whether the loss occurred outside the Center, whether there were missing trajectory predictions, whether there was a common arrival fix (for arrival/arrival pairs only), or whether this conflict was detected in advance. This information is output to a file in simple text output and serves as a quick-reference of loss data for a run.

The second analysis output for loss cases is a detailed plot showing each loss pair. Figure 1 demonstrates a loss-pair plot. This plot is based on the aircraft pair and the loss time, as contained in the loss-pairs file, and shows a 10-minute window before and after the loss time. Instead of using interpolated data, with the one exception of the time of first loss of separation, the data displayed on the plot are the original, non-interpolated, raw data. This is designed to show the data used by CTAS while it was running, as well as any data that might be missing around the time of a loss.

The loss-pair plot consists of two component plots: the upper plot displays horizontal detail and the lower plot displays vertical detail. Some elements are shared between both plots. The legend shows the two aircraft involved: aircraft 1 is ABC123 and aircraft 2 is XYZ456 in this case. Additional data following the aircraft ID indicate the aircraft origin airport, an identification code number, the airplane type, and a code letter indicating onboard equipment. Most data for each aircraft is color-coded with blue for aircraft 1 and green for aircraft 2. In this case green is used without misinterpretation because it is established that this is a loss case. It is also used because it visually complements blue as the second color. Aircraft track data is represented as either X's or O's with an over-size X or O indicating the first track point. Predicted trajectory data is displayed as a series of lines, and is shown for 10 minutes prior to the loss time through 1 minute following the loss time. Trajectory lines often lie close to or on top of each other, so black dots are plotted on the starting point of each trajectory line to help indicate when trajectories may be missing. Aircraft intent data (i.e. flight plan and assigned altitude) is indicated with dashed lines. Simulation control is indicated with an asterisk, either on the plot at the track point where simulation first takes control, such as aircraft 2, or in the legend in front of the aircraft identifier if the aircraft starts in simulation control, such as aircraft 1. Finally there are triangles indicating each minute up to 5 minutes prior to the loss time and red circles to indicate the point of first loss.

Other elements of the two plots are unique to each plot. The upper plot is an X vs. Y plot, while the lower plot is an altitude vs. time plot. Because of this the triangles and loss point help link points between the two plots. The upper plot shows flight plan data, and may include multiple flight plans for each aircraft. To help link the track data with the appropriate flight plan data, the "flown" portions of flight plan data that lie closest to and correspond with the track data are highlighted in a darker color while the rest of the flight plan lines are left in a lighter color. This also means that portions of the flight plan either before or after the 20-minute window of the plot are left in a lighter color. The upper plot also has yellow diamonds in the track data showing where the first conflict detection occurred (if at all), and shows an outline of the ZFW Center boundary to serve as a location reference. The lower plot shows changes in altitude assignment, including temporary altitudes, as instantaneous transitions. Altitude assignments are drawn only in dark lines because they have well-defined beginning and end points, and because the plot shows only the data in the 20-minute window around the loss. The last detail unique to the lower plot is a series of red tick marks along the bottom axis indicating times, if any, when conflicts were predicted for this pair.

For this plot, it was decided that it was important to show all trajectory paths in range before the loss and shortly after it. For an aircraft with 10 full minutes of track data before the loss point, there could be over 50 trajectory predictions in this range. This means over 50 trajectory lines may be drawn for one aircraft. Usually many will overlap each other, and the result is that only a few variations will be seen such as in Fig. 1. Sometimes, though, an aircraft might make a more complicated maneuver and create a series of predictions with differing paths. The resulting plot might be difficult to interpret because the trajectory lines can become confusing.

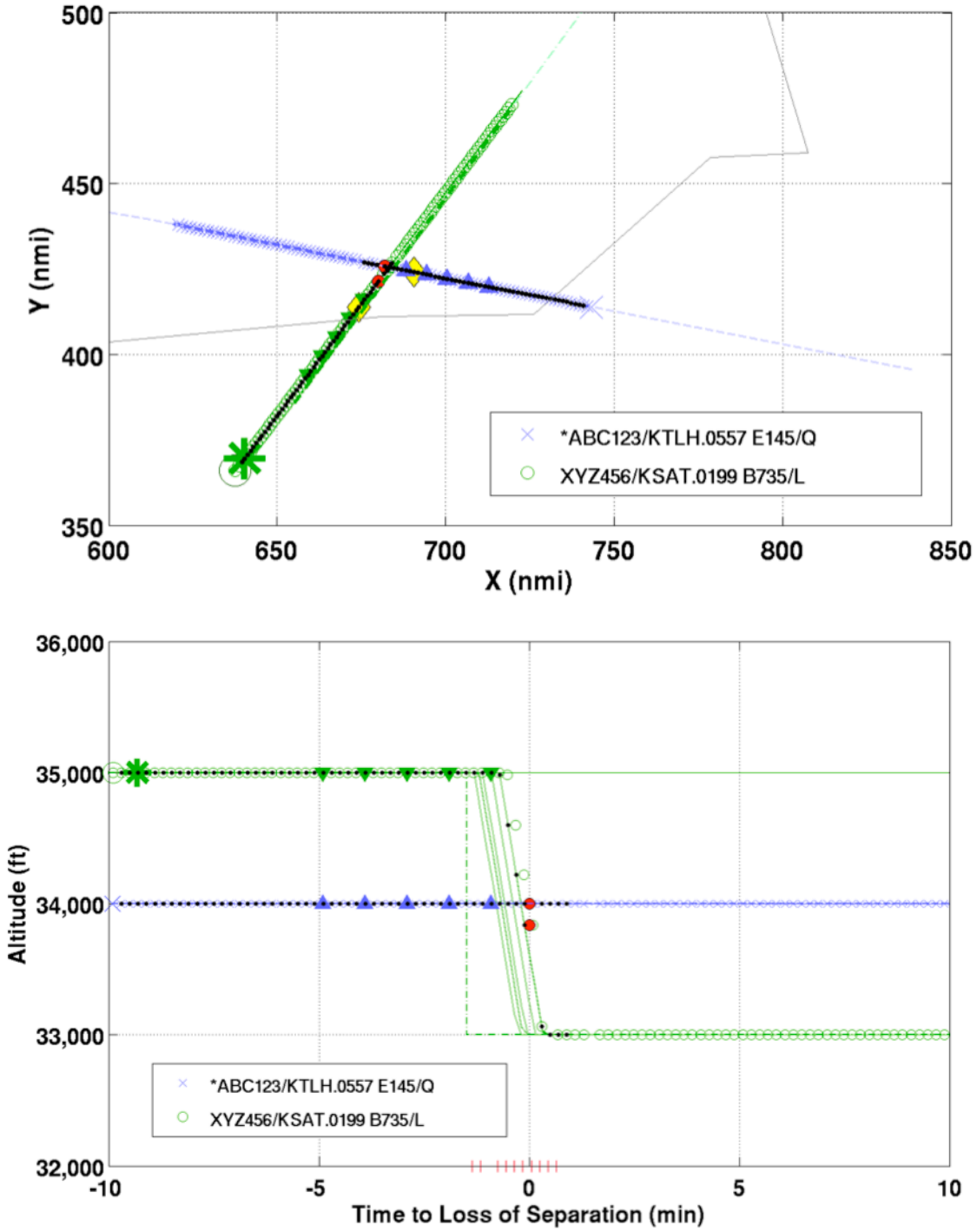

Figure 1. Loss-pair plot

Because of the fact that so many trajectory lines do overlap one another, some additional thought was put into finding a way to show all the different trajectory paths. For a two-dimensional plot, there is no simple way to show so many potentially overlapping lines, and still include all the other existing elements of the plot. One solution might have been to create a series of plots sequenced by time to show all trajectory paths and when they were generated, but this would be a more complex solution and would require additional plots to supplement the loss-pair plot. Instead a compromise was to mark the beginning of each trajectory line with a black dot. Each new trajectory

computation starts from the current track position so the start points don't overlap and create a clear progression of points as the aircraft flies. The result is that while it is often not possible to identify and isolate a single trajectory line from start to end in these plots, any gaps in the progression of start points clearly indicate a failure of trajectory predictions in that region even if surrounding lines are overlapping each other.

The loss-pair plot is useful for displaying a number of data elements pertinent to a loss case in an "at-a-glance" format that allows certain details to stand out clearly. It can show when a loss occurs far outside ZFW Center, when and where track or trajectory data are missing, or when trajectories are perhaps being predicted in an inaccurate or unexpected manner. It also demonstrates the trade-offs and compromises that sometimes need to be made to create an informative and useful plot. For the example of Fig. 1, the upper, X vs. Y plot shows that both aircraft are following their assigned flight plan closely, but, in this case, doesn't provide much more insight on the cause of this loss of separation. The lower altitude vs. time plot, on the other hand, clearly shows that the loss of separation occurred following a change of assigned altitude for the second aircraft. Following this altitude amendment, the conflict was quickly detected as shown by the tick marks on the X-axis, and the trajectory lines suggest that the aircraft was predicted to descend faster than the actual track data shows. Together, these plots suggest that the cause of this loss was either that the aircraft descended slower than predicted, or that the automation failed to properly check the descent path before issuing the change of altitude.

A similar analysis, based on the loss of separation analysis, shows the minimum separation between two specified aircraft. This minimum separation pair analysis is useful for comparing how a pair of aircraft that resulted in a loss case in one run, may have been successfully separated in another run. This type of comparison is most often performed between an open and closed-loop run, or, more commonly, between two different closed-loop runs with different settings or variations in the separation algorithms, but it could also be used to compare against a live run if desired. This analysis is generally used only for select cases of special interest. The plot itself is nearly identical to the loss-pair plot. The main difference is that the reference point is the point of minimum horizontal separation between the two aircraft. Only the horizontal separation is taken into consideration for this plot as the vertical separation between aircraft is often much less than the horizontal separation (minimum legal vertical separation can be as little as 1,000 feet), and usually it is the horizontal separation that is of more interest. The minimum separation point itself could still represent a loss case, and if so, there are two red circles at the minimum separation point in the plot just as in the loss-pair plot. Otherwise the circles are colored green.

B. Conflict Detections

Another metric examines predicted conflict detections during the run. The goal of conflict detection analysis is to help reduce the number of false detections and missed detections, and to ensure that potential loss of separation cases are detected with enough advance warning to create safe, conflict-free, strategic-range resolutions. One difficulty with conflict detection data is that a detection for a given aircraft pair may appear sporadically over several prediction cycles in the archived conflict data set. This can be due to the sensitivity of the prediction settings, uncertainty in predicting the path of an aircraft, or difficulty in computing a trajectory for an aircraft. Because of this inconsistency in detections a study was done in Ref. 6 to examine when the resolution automation should respond to a predicted conflict. The results showed that 3 consecutive detections were a sufficient condition for triggering a resolution call.

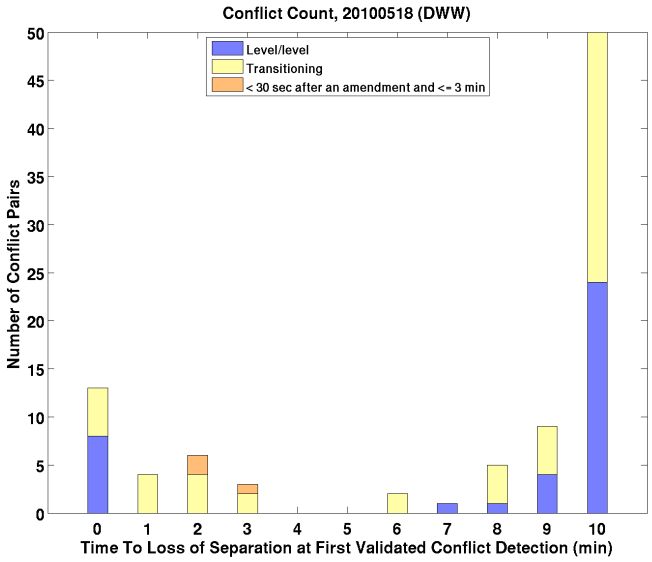

Figure 2. Validated conflict detection metric

The findings of that study formed the basis for the plot in Figure 3. In this plot a conflict is considered "validated" if it is the first of at least three consecutive detections in successive prediction cycles. This may or may not include the initial detection. In addition these counts are divided into two broad groups. Aircraft pairs where both are flying level, level/level pairs, are denoted by blue bars, and pairs where at least one aircraft is climbing or descending, transitioning pairs, are indicated by yellow bars. To highlight

cases where a resolution may have been ineffective or possibly incorrect, an additional group of tactical range conflicts that are detected within 30 seconds after a flight amendment are shown in orange bars. Ideally a resolution amendment should be free of any immediate conflicts so these counts may be cases where an amendment is flawed or the trajectory may not be modeled adequately. The high count at the 10-minute bar is due to the fact that 10 minutes is the maximum range for detecting conflicts, so this reflects conflicts that may be detected farther out, but are not reported until the time to loss reaches 10 minutes.

C. Trajectory Prediction Error

Fundamental to the ability to detect conflicts and compute flight amendments is the algorithm to create predicted trajectories. Thus a few different methods have been developed to analyze predicted trajectories in CTAS. For SA, short-term predictions, those around 5 to 10 minutes out, are of greatest interest for detecting and resolving strategic range conflicts. One metric that is commonly used in SA research is shown in Figures 3, 4, and 5. Although many aircraft trajectories are usually predicted, in an effort to keep this analysis simple, a starting track point and the associated trajectory prediction from that point is selected for an aircraft. The trajectory is then compared to the aircraft's position at some specified time in the future, typically in that 5 to 10 minute range. Because flight amendments may occur at any time, and would introduce larger errors if compared with trajectory predictions made before these amendments were issued, aircraft are filtered to those that have no such amendments, are generally following their flight plan routes, and have enough track data to analyze. While Figs. 3-5 show data from a related set of runs, the number of aircraft analyzed are different for each run because of the results of these filter criteria. This analysis can produce plots to show differences in along-track, cross-track, and altitude predictions. Results are also grouped by departures and overflights. Arrivals would be analyzed based on their top of descent (or ToD) points on final approach to the meter fix, but due to less intent data, the fact that assigned altitudes for aircraft on final approach are not filed, and difficulties in determining the actual ToD point from other level segments and temporary altitudes near the end of the cruise segment, arrivals are more difficult and are not currently included. The plots in Figs. 3-5 show along-track errors for overflight data at 8 minutes out.

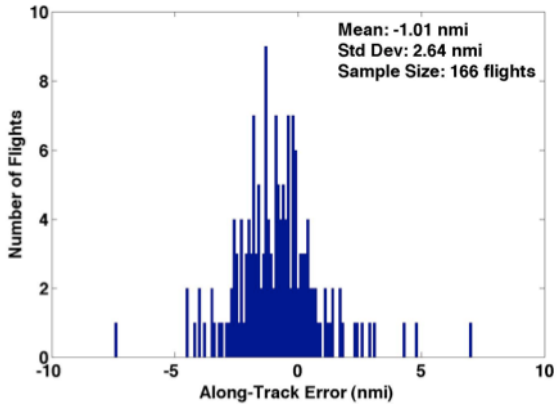

Figure 3. Trajectory prediction error - live run

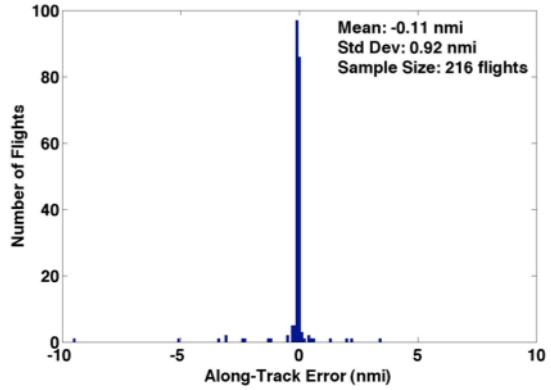

Figure 4. Trajectory prediction error - closed loop

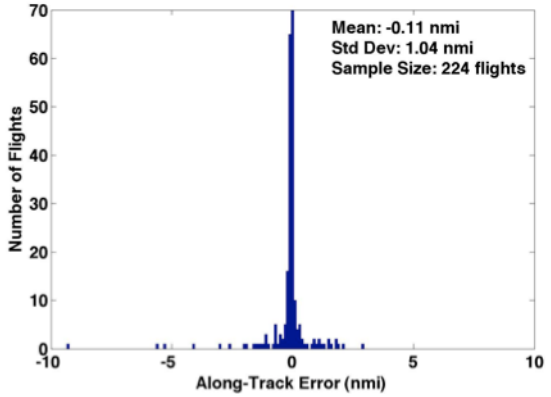

Figure 5. Trajectory prediction error - closed loop with uncertainty

This type of trajectory analysis is performed for both live and simulation runs. When considering trajectory error for live data, there are many factors that can influence an aircraft's flight path including airplane weight, airline company protocols, weather, pilot response time, engine performance, and the pilot's conformance to the assigned path. It is expected there should be significant variation between the predicted and actual flight paths, and this is reflected in the live data result shown in Figure 3. An aircraft in a closed-loop simulation run, on the other hand, is flown based on a single trajectory prediction, and should show close adherence between the predicted and flown paths, as is shown by the sharp center peak with little spread in Figure 4. These two plots demonstrate a difference between simulation results and live data results. This type of analysis shows that in order for simulation results to more closely resemble

live results, it is necessary to model the effects of uncertainty in the flight data. Figure 5 shows the same analysis performed on a different simulation run where some uncertainty has been added in to the aircraft weight and response time as well as to the weather data. For this run there is more spread in the error results, although there is still a strong central peak of little to no error. Because one run offers a relatively small data sample size, this analysis can be combined for multiple data runs to create a much larger data set as is used in Ref. 4.

D. Flight Amendments

One of the goals of SA analysis is to show that the use of automation can simplify the overall management of air traffic. One way of showing this is to examine the number of amendments issued to aircraft. Each amendment that a controller needs to issue to an aircraft may require monitoring of air traffic in the vicinity, formally filing the amendment, contacting the pilot, and waiting for the pilot to accept the amendment. Fewer amendments should correspond to less work for controllers. Figure 6 shows a histogram of counts of different types of amendments issued to aircraft over time for a simulation and the corresponding live run. While these two plots show a striking difference in the overall number of amendments issued between the two runs, these two plots are not measuring exactly the same quantities. This is largely because there is more detail in the simulation amendment data allowing it to be clearly categorized based on what the amendment is and what it is in response to. The live data shows only when route or altitude amendments are made, but not why they were made or what they are in response to. Another difference is that the simulation in this run was set to use altitude amendments only so it shows no use of temporary altitudes. The live results may include many amendments made solely for procedural purposes and not in response to conflicts.[7] As a result, this is not an "apples to apples" comparison, but it is still useful in demonstrating a difference between live data and simulation.

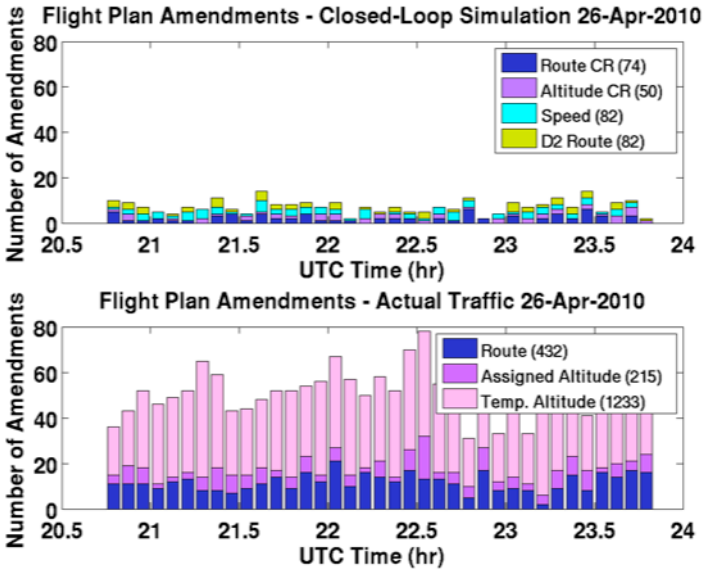

Figure 6. Amendment counts for live vs. simulation

Figure 6 also uses colors to suggest similarities or differences between quantities. The lower plot shows a live data run and includes counts for route changes, assigned altitudes, and temporary altitudes. These are all different types of aircraft path changes and are all represented in different shades of purple. The upper plot is the corresponding simulation run, and here the dark and medium purple colors are used again to represent the similar quantities of route and altitude changes respectively, though in this case, these counts reflect specific conflict resolutions. This same determination cannot easily be made for the live data. As mentioned previously, temporary altitudes are not used in the simulation runs, so these are not represented in the results. There are two additional values, however, in the simulation results. One is a count of speed amendments, and these are indicated using a light blue color. Speed amendments may be used in live data, but they are not indicated and, thus, cannot be counted. Because speed amendments are also used as conflict resolutions, but are not a path change, they are colored light blue to suggest a similarity to the other conflict resolutions colored in purple. The other value is a count of direct-to amendments (denoted as D2 routes). These are not distinguished in the live data. In the simulation data these direct-to routes are not issued as conflict resolutions, as are the other types, but instead are issued to improve traffic flow and reduce flying time. To make these direct-to amendments stand out from the other conflict resolutions, they are colored yellow so that they are a distinctly different color than the other quantities shown on the plot, and the direct-to bars are positioned on top of the stack.

The simulation data for amendments also includes delta time information based on the difference between an aircraft's original flight path, and its flight path after an amendment. These delta time values allow a measurement of how well the automation is computing amendment flight paths. In general, a conflict resolution should minimize flight delays, and may sometimes result in a small time savings if the amendment involves flying direct to a downstream fix. If the simulation run includes automated direct-to amendments (not restricted to conflict

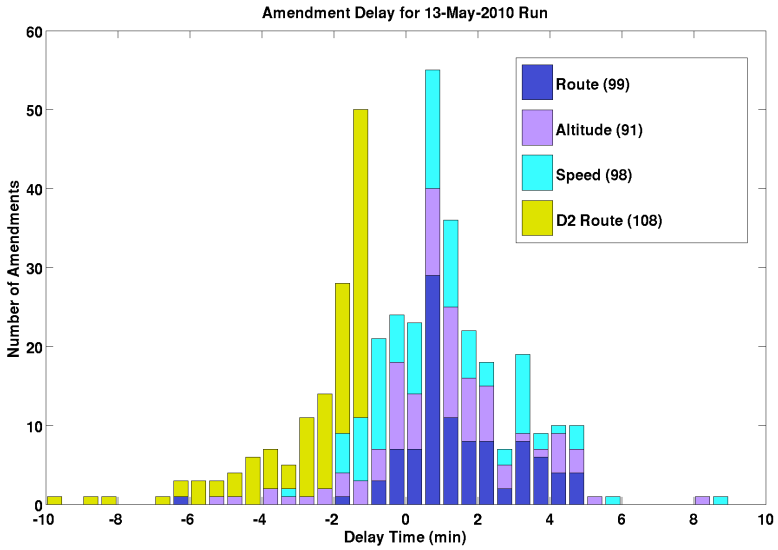

Figure 7. Closed-loop delta time histogram

resolutions), then these direct-to amendments should generally reflect only small time savings as the direct-to algorithm limits the range of downstream fixes that can be used.[4] Figure 7 shows a plot of these delta time values expressed as delay time in 30-second bins and plotted vs. the number of amendments (note that in this representation, time savings is plotted as a negative value). This plot uses the same color scheme as the closed-loop plot in Figure 6 because the data is related and the same types of amendments are plotted (although in this paper, Figure 6 and Figure 7 were chosen for illustrative purposes and do not reflect the same data set). Currently the plot in Figure 7 is created only for closed-loop simulation cases. Changes to CTAS will allow for similar delta time data to be created for both live and closed-loop runs thus allowing for a direct comparison between them.

E. Level-Offs

Another method that provides a measure of controller workload, as well as flight path optimization, is to examine aircraft level-offs during aircraft departures or arrivals. Level-offs are commonly used to resolve conflicts and also as a part of standardized departure or arrival procedures. At the same time each level-off issued requires a controller action to initiate, and possibly end, and interrupts a climb or descent segment potentially resulting in greater fuel usage especially for descending aircraft.

In the data, some of these level-offs may be recorded as temporary altitudes, but often, many are not, and some temporary altitudes may be modified or changed before an aircraft reaches them, so they do not even result in a corresponding level-off. As a result the only way to measure level-offs is to identify them from the aircraft track data. In order to do this, specific criteria need to be applied to determine what kind of data constitutes a level-off, and exactly where it begins and ends.

A few methods were tried, but one that seemed to work best involved examining the track data just three consecutive points at a time. First the difference between each successive altitude track value is computed to determine whether the aircraft is climbing, descending or level at each point. If there are no more than two points of climbs or descents in the set of three and an altitude difference of no more than 200 feet between the first and last point, the aircraft is considered to be level. Where both conditions are first satisfied is where the segment begins, and where either is violated, the segment ends. This can allow a slight climb or descent on the ends of a level segment, but it tends to keep these to a minimum while still retaining relevant data. It also allows the aircraft's altitude to vary slightly during a level segment, but this is consistent, especially with live data, where aircraft may not maintain a constant altitude value throughout level flight segments. An additional constraint is added that a level segment must be about a minute long. This eliminates very short segments that are not of interest.

Figure 8 shows the result of these conditions applied to arrival data from a live run. Arrival tracks from the northeast quadrant of the Center are plotted as altitude vs. simulation time with thin lines, and level-offs, after an initial descent, are highlighted with a thicker line. An additional altitude cutoff of 11,000 feet was applied to remove portions of tracks in TRACON airspace. Once identified, these level-offs can be counted and their durations measured for additional analysis. The title of the plot, for example, shows a count of detected level-offs as well as their average duration. When this same analysis was applied to a closed-loop run for this same data set, only 1 level-off was detected, and in fact over similar closed-loop runs typically there are either no level-offs or only a few so simply comparing a plot such as Figure 8 between live and simulation runs again shows a large difference between live and simulation. Although the plot in Figure 8 shows arrivals, a similar analysis may be done for departures.

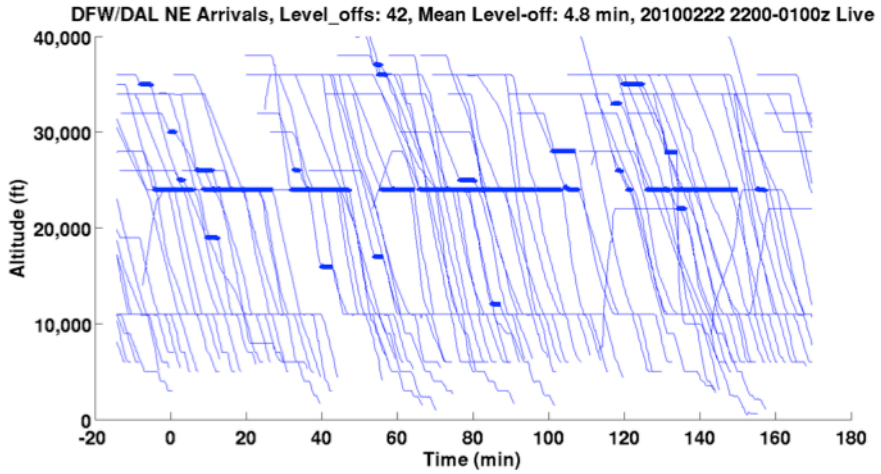

Figure 8. Altitude track data from live run with level-offs highlighted

F. Fuel Burn

A final metric that has been developed, but is still undergoing revision, is a measurement of fuel use. Fuel consumption has long been an important concern to airline companies so it is useful to show what effect automation results have on fuel usage. Unfortunately this is difficult to accomplish for a number of reasons. Fuel used depends on a number of factors including the aircraft's flight phase (climbing, level or descending), aircraft mass, speed, altitude, and thrust among others. Aircraft mass, in particular, is not available in the data set, and must be assumed based on range limits for each aircraft. This means that at best, fuel use may only be approximated. The problem is further complicated when factoring in the rate of change of aircraft mass as fuel is burned, and the change in fuel burn rates for different altitudes, speeds, or flight phase.

In order to try to simplify the problem to provide some ballpark estimates of fuel use, an approach was tried using only overflights in order to eliminate long climbs or descents. Each aircraft's flight path was restricted to the portion of the flight through ZFW Center airspace. The mass was assumed to be constant during the flight, and each aircraft was assumed to travel at a specified constant speed and altitude. The altitude was selected as the assigned altitude of longest duration during the flight path, and the speed chosen was averaged over the course of the flight path. Both values were selected from the live data run only. An initial attempt at this analysis used the difference in times between the Center entry and exit points as the duration of the flight. These values were used along with BADA[9] models providing average aircraft mass, thrust, and fuel burn rates to compute fuel burn for the live run.

For the simulation data, the idea was to compute the fuel usage assuming the same altitude and average speed of the live data, but using the paths flown in the simulation run. Thus a list of aircraft processed from the live run along with the associated speed and altitude used was passed as input to the analysis of the simulation run. The aircraft flight paths were computed for overflight data in the simulation run, and the duration of each flight was computed using the distance from the simulation run and the speed from the live run. Using these values the fuel use for the simulation run could then be computed. However, because of differences between each run's progression, not all aircraft processed the first time from the live run, will be processed in the simulation run. The aircraft processed in the simulation run, on the other hand, are restricted to the same aircraft from the live run, so no new aircraft will be introduced by analyzing the simulation run. To account for this difference, a list of aircraft processed by the simulation run is passed back as input for a secondary analysis of the live run. This way the results of the live run are filtered to a common set of aircraft for both the live and simulation runs.

When the results of this analysis were examined, they showed that the simulation run used more fuel, but flew fewer miles than the live run. Upon closer examination it was noticed that when the average speed was used with the flight path distance to compute the flight duration in the live data, as was applied to the simulation data, instead of using the Center entry and exit times, the result was that flight durations were computed to be slightly longer. Correspondingly, this should result in a higher fuel usage for the live data. Although this approach creates an artificial flight duration time for the live data, it does treat live and simulation data the same way and provides a closer comparison between the two. The analysis was modified accordingly and rerun. The results this time showed that the simulation run had both fewer miles and less fuel consumption, though the difference was small. Figure 9 shows a plot of the overflight tracks across the Center airspace for a live run with a dot indicating the Center entry point and an X indicating a Center exit point to give an idea what the track data looks like across the Center and what portion is being used for the fuel analysis. The number of aircraft, miles flown in Center, and corresponding computed fuel use associated with these aircraft are all indicated in the plot title.

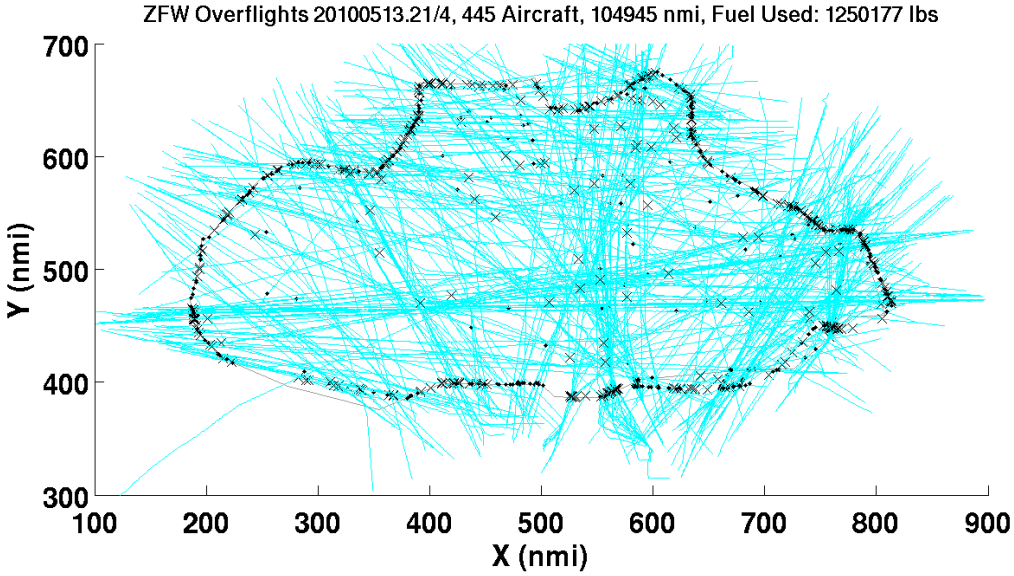

Figure 9. Overflight track data from live run over ZFW Center

Further analysis on these results divided the aircraft by airline and aircraft types, and this showed that some types of aircraft still had notably increased fuel consumption in the simulation data. By examining the track data for some specific aircraft, and comparing them between the runs, it was observed that in some cases, an aircraft in the live run might take a short path near the boundary or fly north or south across the short axis of ZFW Center and then pass into a neighboring Center and eventually out of data range. In the corresponding simulation run, this same aircraft would start with the same initial path, but then take a much longer path across the full width of the Center after receiving an amendment such as a direct-to. While the overall path of the aircraft might be shorter in the simulation run, the time spent in ZFW and the corresponding fuel use in the Center would be much longer than in the live run thus distorting the analysis results. For now this analysis remains primarily experimental until a better solution is found to handle cases of significant path changes such as these. In addition, fuel computations for departure and arrival aircraft would also be desirable, although these cases are more difficult to compute.

VII. Concluding Remarks

Useful plots and metrics are sometimes the result of specific research or software development needs, but other times are the result of an iterative process based on refining research goals or evolving software capabilities or issues. Some can even be the result of interesting or unexpected results discovered as part of another, potentially unrelated, analysis. This paper has documented several examples of the metrics and types of analysis used in SA research at NASA to study both live and simulated air traffic scenarios in terms of loss of separation, conflict detection, trajectory prediction error, flight amendments, level-offs, and fuel burn and the process by which these metrics and analysis plots were created. Some of these metrics are designed to look at aggregate properties of an entire run such as the analysis of fuel consumption, and others are designed to focus in on specific details within a run such as the loss-pair plot. Whether such plots and metrics support a research project directly, such as being published in a scientific paper or study, or indirectly, such as being used as part of a routine set of software tests, they are a crucial element of the research process.

Acknowledgments

The author would like to thank Dave McNally of NASA for advice and suggestions for this paper as well as for discussions on plot design, air traffic management techniques, and aircraft performance characteristics. The author would also like to thank Neil Chen of NASA for discussion on using BADA data tables as well as supplying a function to access them in Matlab for fuel burn computations. Thanks are also due to Jinn-Hwei Cheng of the University of California at Santa Cruz for performing the numerous data runs on which this research is based.

References

[1]McNally, D., and Gong, C., "Concept and Laboratory Analysis of Trajectory-Based Automation for Separation Assurance," AIAA-2006-6600, AIAA Guidance, Navigation and Control Conference and Exhibit, Keystone, CO, 21-24 Aug. 2006.

[2]Erzberger, H., Davis, T. J., Green, S. M., "Design of Center-TRACON Automation System," Proceedings of the AGARD Guidance and Control Panel 56th Symposium on Machine Intelligence in Air Traffic Management, Berlin GDR, 1993.

[3]Murphy, J., Robinson, J., "Design of a Research Platform for En Route Conflict Detection and Resolution," Proc. AIAA Aviation Technology Integration and Operations Conference, AIAA-2007-7803, Belfast, Northern Ireland, 2007.

[4]Erzberger, H., McNally, D., Foster, M., Chiu, D., Stassart, P., "Direct-to Tool for En Route Controllers," ATM99: IEEE Workshop on Advanced Technologies and their Impact on Air Traffic Management in the 21st Century, Capri, Italy, Sept. 26-30, 1999.

[5]Erzberger, H., "Automated Conflict Resolution for Air Traffic Control", *Proceedings of the 25th International Congress of the Aeronautical Sciences (ICAS)*, Hamburg, Germany, 2006.

[6]McNally, D., Thipphavong, D., "Automated Separation Assurance in the Presence of Uncertainty", *Proceedings of the 26th International Congress of the Aeronautical Sciences (ICAS)*, Anchorage, Alaska, USA, 2008.

[7]McNally, D., Mueller, E., Thipphavong, D., "A Near-Term Concept for Trajectory-Based Operations with Air/Ground Data Link Communication," Proceedings of ICAS 2010 27th International Congress of the Aeronautical Sciences, Nice, France, Sept. 19-24, 2010.

[8]Paielli R., Erzberg, H., Chiu, D., "Tactical Conflict Alerting Aid for Air Traffic Controllers," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 32, No.1, Jan.-Feb. 2009.

[9]Nuic, A., "Base of Aircraft Data (BADA) Product Management Document," EUROCONTROL Experimental Center, EEC Technical/Scientific Report No. 2009/008, France, March, 2009.

[10]Arend, L., Logan, A., Havin, G., *Using Color in Information Display Graphics*, "Designing a Color Graphics Page" [online user reference], URL: http://colorusage.arc.nasa.gov/graphics_page_design.php [cited 4 February 2010].

[11]MATLAB, Software Package, Ver. 7.10.0.499 (R2010a), The Mathworks, Natick, MA, 2010.